

**Data Technician**

|  |
| --- |
|  |

|  |
| --- |
| Name: |
| Course Date: |
|  |

**Table of contents**

[Day 1: Task 1 3](#_Toc106466110)

[Day 1: Task 2 3](#_Toc1673105856)

[Day 3: Task 1 4](#_Toc1682646377)

[Day 4: Task 1: Written 6](#_Toc135095808)

[Day 4: Task 2: SQL Practical 9](#_Toc1228298484)

[Course Notes 19](#_Toc631049525)

[Additional Information 19](#_Toc1612679949)

# Day 1: Task 1

Please research and complete the below questions relating to key concepts of databases.

|  |  |
| --- | --- |
| What is a primary key? | A primary key in a database is a column (or a collection of columns) that uniquely identifies each record (row) in a table. It ensures that no two rows have the identical value(s) in that column(s), hence preserving data integrity.  Here are the main points to consider a primary key:   * Uniqueness: Each value in the primary key column(s) must be unique throughout the table. * Non-null: A primary key cannot contain NULL values since each row must have a value in order to be uniquely identified. * A primary key can be a single column (simple key) or a combination of columns (composite key) that uniquely identifies a row. * Indexing: Most database systems generate an index on the primary key to speed up searches and queries. |
| How does this differ from a secondary key? | A primary key is a column or set of columns in a database table that uniquely identifies each record and cannot have any null values, maintaining data integrity inside the table. A secondary key, on the other hand, is used to access or organise data but does not have to be unique and may contain duplicates; it is frequently used to establish relationships between tables, such as a foreign key which links one table to another. While the primary key ensures uniqueness and identification, the secondary key primarily functions as a reference or search tool. |
| How are primary and foreign keys related? | A primary key and a foreign key share similarities since they define relationships between tables in a database. The primary key uniquely identifies each record in its own table, whereas a foreign key is a column (or combination of columns) in another table that refers to the primary key. This relationship maintains referential integrity. |
| Provide a real-world example of a one-to-one relationship | **Person → Passport**   * Each person can have exactly one passport. * Each passport is assigned to exactly one person.   **Employee → Employee Badge**   * Each employee is issued exactly one badge. * Each badge is assigned to exactly one employee.   **Student → Student ID Card**   * Each student gets one ID card. * Each ID card belongs to one student.   **Country → Capital City**   * Each country has exactly one capital city. * Each capital city belongs to exactly one country.   **Driver → Driver’s License**   * Each driver can have one driver’s licence. * Each licence is issued to one driver only. |
| Provide a real-world example of a one-to-many relationship | **Customer → Orders**   * Each customer can have multiple orders. * Each order belongs to exactly one customer.   **Author → Books**   * Each author can write multiple books. * Each book is written by exactly one author.   **Teacher → Students**   * Each teacher can teach multiple students. * Each student has exactly one teacher.   **Company → Employees**   * Each company can have multiple employees. * Each employee works for exactly one company.   **Library → Books**   * Each library can have multiple books. * Each book belongs to exactly one library.   **Country → Cities**   * Each country can have multiple cities. * Each city belongs to exactly one country. |
| Provide a real-world example of a many-to-many relationship | **Students ↔ Courses**   * Each student can enrol in multiple courses. * Each course can have multiple students.   **Actors ↔ Movies**   * Each actor can act in multiple movies. * Each movie can have multiple actors.   **Doctors ↔ Patients**   * Each doctor can treat multiple patients. * Each patient can be treated by multiple doctors.   **Authors ↔ Magazines**   * Each author can write for multiple magazines. * Each magazine can have articles by multiple authors. * **Teachers ↔ Subjects** * Each teacher can teach multiple subjects. * Each subject can be taught by multiple teachers. * **Companies ↔ Projects** * Each company can work on multiple projects. * Each project can involve multiple companies. |

# Day 1: Task 2

Please research and complete the below questions relating to key concepts of databases.

|  |  |
| --- | --- |
| What is the difference between a relational and non-relational database? | A relational database (RDBMS) organises data into structured tables with rows and columns, where each table has a fixed schema and relationships are enforced using keys. It uses SQL to query and manage data and is ideal for applications requiring strict consistency, like banking or enterprise systems. In contrast, a non-relational database (NoSQL) stores data in flexible formats such as documents, key-value pairs, or graphs, allowing dynamic schemas and easier horizontal scaling. NoSQL databases are better suited for applications requiring high performance and handling large, unstructured datasets, like social media platforms or real-time analytics systems.  **Structure**  * Relational Database (RDBMS):   + Stores data in tables (rows and columns).   + Schema is fixed: every table has a defined structure.   + Example: MySQL, PostgreSQL, Oracle. * Non-Relational Database (NoSQL):   + Stores data in flexible formats: documents, key-value pairs, graphs, or wide-column stores.   + Schema is dynamic: fields can vary across records.   + Example: MongoDB, Redis, Cassandra.  **Relationships**  * RDBMS:   + Relationships are defined using foreign keys.   + Supports joins to combine data from multiple tables. * NoSQL:   + Relationships are usually embedded or referenced but not enforced by the database.   + Typically avoids joins; optimised for fast reads/writes.  **Query Language**  * RDBMS: Uses SQL (Structured Query Language). * NoSQL: Uses varied query methods, often specific to the database (e.g., MongoDB uses JSON-like queries).  **Scalability**  * RDBMS: Usually vertically scalable (scale-up: more powerful server). * NoSQL: Usually horizontally scalable (scale-out: add more servers).  **Use Cases**  * RDBMS:   + Banking systems, ERP, and transactional systems—where data consistency is critical. * NoSQL:   + Social media, IoT, and real-time analytics—where high performance and flexible schemas matter. |
| What type of data would benefit off the non-relational model?  Why? | Non-relational (NoSQL) databases are best suited for large-scale, unstructured, or rapidly changing data.  Examples include:   * Social media content – includes posts, comments, likes, and multimedia, which vary in form and develop quickly. * IoT sensor data - consists of time-stamped readings that may not adhere to a particular structure. * E-commerce product – catalogues contain products with various attributes, classifications, and specifications. * Real-time analytics – clickstream data or event logs that require quick insertion and querying.   Why:   * NoSQL databases allow flexible schemas, which means you don't have to predefine every column. * They enable horizontal scaling, which efficiently distributes large amounts of data across several servers. * They enable fast read and write speeds, making them excellent for real-time applications. * They can hold a variety of data kinds (JSON, key-value, and graph), making them suitable for a wide range of datasets. |

# Day 3: Task 1

Please research the below ‘JOIN’ types, explain what they are and provide an example of the types of data it would be used on.

|  |  |
| --- | --- |
| Self-join | **SELF JOIN**  * **What it is:** A join where a table is joined with itself. Useful for comparing rows within the same table. * **Use case:** Find hierarchical or relational data within a single table, like managers and employees. * **Example Data:** Employees table to find which employees report to which manager.   **SELECT e1.name AS employee, e2.name AS manager**  **FROM employees e1**  **LEFT JOIN employees e2**  **ON e1.manager\_id = e2.id;** |
| Right join | **RIGHT JOIN (or RIGHT OUTER JOIN)**  * **What it is:** Returns all rows from the right table, and matching rows from the left table. If there’s no match, the left table columns are NULL. * **Use case:** Find all departments and include employees if assigned. * **Example Data:** Departments table (all departments) and Employees table.   **SELECT e.name, d.department\_name**  **FROM employees e**  **RIGHT JOIN departments d**  **ON e.department\_id = d.id;** |
| Full join | **FULL JOIN (or FULL OUTER JOIN)**  * **What it is:** Returns all rows from both tables, with NULL where there is no match. * **Use case:** Combine two datasets completely, including unmatched rows. * **Example Data:** Employees and Contractors tables to get all people working in a company.   **SELECT e.name, c.contractor\_name**  **FROM employees e**  **FULL OUTER JOIN contractors c**  **ON e.id = c.employee\_id;** |
| Inner join | **INNER JOIN**  * **What it is:** Returns only the rows where there is a match in both **tables**. * **Use case:** Useful when you only want related data from both tables. * Example Data: Employees and Departments. Only return employees assigned to a department.   **SELECT e.name, d.department\_name**  **FROM employees e**  **INNER JOIN departments d**  **ON e.department\_id = d.id;** |
| Cross join | **CROSS JOIN**  * **What it is:** Returns the **Cartesian product** of two tables (every row from table A paired with every row from table B). * **Use case:** Generate all possible combinations, such as product bundles or schedules. * **Example Data:** Products table and Colors table to generate all product-color combinations.   **SELECT p.product\_name, c.color**  **FROM products p**  **CROSS JOIN colors c;** |
| Left join | **LEFT JOIN (or LEFT OUTER JOIN)**  * **What it is:** Returns all rows from the left table and matching rows from the right table. If there’s no match, the right table columns are NULL. * **Use case:** Find all employees and include department info if available. * **Example Data:** Employees table (all employees) and Departments table.   **SELECT e.name, d.department\_name**  **FROM employees e**  **LEFT JOIN departments d**  **ON e.department\_id = d.id;** |

# Day 4: Task 1: Written

In your groups, discuss and complete the below activity. You can either nominate one writer or split the elements between you. Everyone however must have the completed work below:

*Imagine you have been hired by a small retail business that wants to streamline its operations by creating a new database system. This database will be used to manage inventory, sales, and customer information. The business is a small corner shop that sells a range of groceries and domestic products. It might help to picture your local convenience store and think of what they sell. They also have a loyalty program, which you will need to consider when deciding what tables to create.*

*Write a 500-word essay explaining the steps you would take to set up and create this database. Your essay should cover the following points:*

1. ***Understanding the Business Requirements****:*
   1. *What kind of data will the database need to store?*
   2. *Who will be the users of the database, and what will they need to accomplish?*
2. ***Designing the Database Schema****:*
   1. *How would you structure the database tables to efficiently store inventory, sales, and customer information?*
   2. *What relationships between tables are necessary (e.g., how sales relate to inventory and customers)?*
3. ***Implementing the Database****:*
   1. *What SQL commands would you use to create the database and its tables?*
   2. *Provide examples of SQL statements for creating tables and defining relationships between them.*
4. ***Populating the Database****:*
   1. *How would you input initial data into the database? Give examples of SQL INSERT statements.*
5. ***Maintaining the Database****:*
   1. *What measures would you take to ensure the database remains accurate and up to date?*
   2. *How would you handle backups and data security?*

*Your essay should include specific examples of SQL commands and explain why each step is necessary for creating a functional and efficient database for the retail business.*

|  |  |
| --- | --- |
| Please write your 500-word essay here | **Setting Up a Retail Database System** Creating a database for a small retail shop involves several clear steps: understanding requirements, designing the schema, implementing it in SQL, populating it with data, and maintaining it to ensure accuracy and security. **Understanding the Business Requirements** The shop sells groceries and household items, so the database must store **product details, stock levels, sales records, and customer information**. Customers are enrolled in a **loyalty scheme**, which means loyalty points must be tracked and updated with every purchase.  The main users of the database will be:   * **Sales staff** – record transactions and update stock. * **Shop managers** – analyse sales trends and customer loyalty data. * **The owner** – monitors shop performance. * **IT staff** – maintain the system, manage access rights, and ensure security.   This database will allow the shop to keep stock current, reward loyalty, make data-driven decisions, and protect customer information.  ***2. Designing the Database Schema:***  The database schema is designed using three core tables—Products, Customers, and Sales—to efficiently manage inventory, transactions, and loyalty information.  Entity- Relation Diagram    Structure:    **Relationships Between Tables**  - Products → Sales: One-to-many (a product can appear in many sales). - Customers → Sales: One-to-many (a customer can make many purchases). These relationships ensure that sales are properly linked to both products and customers. Database Schema Type  * The design uses a Star Schema:   + Central fact table: Sales (records transactions).   + Dimension tables: Products, Customers (describe entities).   + Advantage: Optimized for querying sales, customer analysis, and reporting.  Key Selection: Surrogate vs Natural Key  * Surrogate Key: Used for ProductID, CustomerID, SaleID.   + Numeric IDs (INT) are compact, fast for indexing, and stable over time.   + Avoids problems with natural keys  **Implementing the Database** Once the schema is planned, SQL can be used to create the database and tables by creating a new database, defining columns such as Product Name, Price, and Stock Level, and then adding sample rows like Milk (£1.20, 50 units) or Bread (£1.00, 30 units).  Example:  CREATE DATABASE RetailStore;  USE RetailStore;    CREATE TABLE Products (  ProductID INT PRIMARY KEY AUTO\_INCREMENT,  ProductName VARCHAR(50),  Category VARCHAR(50),  Price DECIMAL(5,2),  StockLevel INT  );    CREATE TABLE Customers (  CustomerID INT PRIMARY KEY,  Name VARCHAR(50),  Email VARCHAR(100),  LoyaltyPoints INT  );    CREATE TABLE Sales (  SaleID INT PRIMARY KEY,  ProductID INT,  CustomerID INT,  Date DATE,  Quantity INT,  );  *Then use foreign keys to link tables*  *FOREIGN KEY (ProductID) REFERENCES Products(ProductID),*  *FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)* **Populating the Database** Initial data can be added manually with **INSERT** statements, or imported from spreadsheets (CSV).  Example:  INSERT INTO Products (ProductName, Category, Price, StockLevel)  VALUES ('Milk', 'Dairy', 1.50, 10),  ('Bread', 'Bakery', 1.00, 20);    INSERT INTO Customers (CustomerID, Name, Email, LoyaltyPoints)  VALUES (12345678, 'Jane Doe', 'janedoe@example.com', 50);    INSERT INTO Sales (SaleID, ProductID, CustomerID, Date, Quantity)  VALUES (12345, 1, 12345678, '2025-09-25', 2); **Maintaining the Database** To remain useful, the database must be **regularly updated**. For example, stock can be reduced after a sale:  Other maintenance measures include:   * **Backups**: full weekly backups and daily differential backups. * **Data security**: staff roles determine access (sales staff record sales, managers update products, IT revoke access when staff leave). * **Error checking**: removing duplicates and filling missing data.  **Conclusion** By following these steps—requirements analysis, schema design, implementation, population, and maintenance—the shop gains a functional and efficient database. It will support stock management, customer loyalty rewards, data-driven insights, and strong data security, all of which are vital to the success of a small retail business. |

# Day 4: Task 2: SQL Practical

In your groups, work together to answer the below questions. It may be of benefit if one of you shares your screen with the group and as a team answer / take screen shots from there.

**Setting up the database:**

1. **Download world\_db(1)**
2. **Follow each step to create your database**

**For each question I would like to see both the syntax used and the output.**

1. **Count Cities in USA:** *Scenario:* You've been tasked with conducting a demographic analysis of cities in the United States. Your first step is to determine the total number of cities within the country to provide a baseline for further analysis.

|  |
| --- |
|  |

1. **Country with Highest Life Expectancy:** *Scenario:* As part of a global health initiative, you've been assigned to identify the country with the highest life expectancy. This information will be crucial for prioritising healthcare resources and interventions.

|  |
| --- |
|  |

1. **"New Year Promotion: Featuring Cities with 'New :** *Scenario:* In anticipation of the upcoming New Year, your travel agency is gearing up for a special promotion featuring cities with names including the word 'New'. You're tasked with swiftly compiling a list of all cities from around the world. This curated selection will be essential in creating promotional materials and enticing travellers with exciting destinations to kick off the New Year in style.

|  |
| --- |
|  |

1. **Display Columns with Limit (First 10 Rows):** *Scenario:* You're tasked with providing a brief overview of the most populous cities in the world. To keep the report concise, you're instructed to list only the first 10 cities by population from the database.

|  |
| --- |
|  |

1. **Cities with Population Larger than 2,000,000:** *Scenario:* A real estate developer is interested in cities with substantial population sizes for potential investment opportunities. You're tasked with identifying cities from the database with populations exceeding 2 million to focus their research efforts.

|  |
| --- |
|  |

1. **Cities Beginning with 'Be' Prefix:** *Scenario:* A travel blogger is planning a series of articles featuring cities with unique names. You're tasked with compiling a list of cities from the database that start with the prefix 'Be' to assist in the blogger's content creation process.

|  |
| --- |
|  |

1. **Cities with Population Between 500,000-1,000,000:** *Scenario:* An urban planning committee needs to identify mid-sized cities suitable for infrastructure development projects. You're tasked with identifying cities with populations ranging between 500,000 and 1 million to inform their decision-making process.

|  |
| --- |
|  |

1. **Display Cities Sorted by Name in Ascending Order:** *Scenario:* A geography teacher is preparing a lesson on alphabetical order using city names. You're tasked with providing a sorted list of cities from the database in ascending order by name to support the lesson plan.

|  |
| --- |
|  |

1. **Most Populated City:** *Scenario:* A real estate investment firm is interested in cities with significant population densities for potential development projects. You're tasked with identifying the most populated city from the database to guide their investment decisions and strategic planning.

|  |
| --- |
|  |

1. **City Name Frequency Analysis: Supporting Geography Education** *Scenario*: In a geography class, students are learning about the distribution of city names around the world. The teacher, in preparation for a lesson on city name frequencies, wants to provide students with a list of unique city names sorted alphabetically, along with their respective counts of occurrences in the database. You're tasked with this sorted list to support the geography teacher.

|  |
| --- |
|  |

1. **City with the Lowest Population:** *Scenario:* A census bureau is conducting an analysis of urban population distribution. You're tasked with identifying the city with the lowest population from the database to provide a comprehensive overview of demographic trends.

|  |
| --- |
|  |

1. **Country with Largest Population:** *Scenario:* A global economic research institute requires data on countries with the largest populations for a comprehensive analysis. You're tasked with identifying the country with the highest population from the database to provide valuable insights into demographic trends.

|  |
| --- |
|  |

1. **Capital of Spain:** *Scenario:* A travel agency is organising tours across Europe and needs accurate information on capital cities. You're tasked with identifying the capital of Spain from the database to ensure itinerary accuracy and provide travellers with essential destination information.

|  |
| --- |
|  |

1. **Country with Shortest Life Expectancy:** *Scenario:* A healthcare foundation is conducting research on global health indicators. You're tasked with identifying the country with the highest life expectancy from the database to inform their efforts in improving healthcare systems and policies.

|  |
| --- |
| - |

1. **Cities in Europe:** *Scenario:* A European cultural exchange program is seeking to connect students with cities across the continent. You're tasked with compiling a list of cities located in Europe from the database to facilitate program planning and student engagement.

|  |
| --- |
|  |

1. **Average Population by Country:** *Scenario:* A demographic research team is conducting a comparative analysis of population distributions across countries. You're tasked with calculating the average population for each country from the database to provide valuable insights into global population trends.

|  |
| --- |
|  |

1. **Capital Cities Population Comparison:** *Scenario:* A statistical analysis firm is examining population distributions between capital cities worldwide. You're tasked with comparing the populations of capital cities from different countries to identify trends and patterns in urban demographics.

|  |
| --- |
|  |

1. **Countries with Low Population Density:** *Scenario:* An agricultural research institute is studying countries with low population densities for potential agricultural development projects. You're tasked with identifying countries with sparse populations from the database to support the institute's research efforts.

|  |
| --- |
|  |

1. **Cities with High GDP per Capita:** *Scenario:* An economic consulting firm is analysing cities with high GDP per capita for investment opportunities. You're tasked with identifying cities with above-average GDP per capita from the database to assist the firm in identifying potential investment destinations.

|  |
| --- |
|  |

1. **Display Columns with Limit (Rows 31-40):** *Scenario:* A market research firm requires detailed information on cities beyond the top rankings for a comprehensive analysis. You're tasked with providing data on cities ranked between 31st and 40th by population to ensure a thorough understanding of urban demographics.

|  |
| --- |
|  |

|  |
| --- |
| **Course Notes** |

It is recommended to take notes from the course, use the space below to do so, or use the revision guide shared with the class:

|  |
| --- |
| **Additional Information** |

We have included a range of additional links to further resources and information that you may find useful, these can be found within your revision guide.

**END OF WORKBOOK**

**Please check through your work thoroughly before submitting and update the table of contents if required.**

**Please send your completed work booklet to your trainer.**